Exhibit E

A Portion of the Documentation for Build 67 of WebBase

Home
Products
WebBase
ExperForms
Consulting
Contact
News

Newest Release Notes

Back

Build 67

Release Notes, September 2001

This brief overview will describe the upgrades and features in **Builds 53 through 67** so you will get the greatest benefit out of this release.

The fixes and new features incorporated in this release have been in use for some time - both internally and in some very large commercial web sites - to allow us to provide you with the best possible product.

Please review this release information and address any questions to our **Support Forum** or email our **Support Staff.**

Contents

    - Current **Auto-Default Functions**
        - date **(Build 62)**
        - time **(Build 62)**
- **Special Forms Processing (Build 26)**
    - Normal, default forms processing
    - New "directory based" forms processing
        - Processing all forms
        - Skipping all forms
        - Overriding a form's mime type
    - The registry **Specials** key
- **Y2K**
    - Controlling **default date displays**
    - Accepting **2-digit year input** as year-2000 based
    - **asShort** and **asLong** methods
- **Domains**
    - Addressing domains via the **Host** header parameter
    - New **Domain Variables**
        - New **SetDomain** and **RemoveDomain** macros
    - **domain** and **domain:** methods on **%cmd%**
    - Accessing other domains
    - **Default Domain**
    - **System Registry**
- **Error Wrappers**
    - Trapping **WebBase scripting** errors
    - The **Error000** Log Files
    - Overriding the **Error000** wrapper/log files
    - Related Variables
    - General **Server Status** messages (403, 404, ...)
- **File Uploads**
    - Multipart FORM post
    - **WebBase** %cmd% contents
    - Remote Web Site Maintenance
    - Performance Issues
- **BrowserFlush Macro**
    - Streaming Output
    - **BrowserFlush** Macro
    - Display Considerations
    - Performance Considerations
- **New or modified Variables**
- **New Functions/Macros**
- **Bug Fixes and Upgrades**
    - Compatibility with latest **ODBC Drivers**
        Addresses the errors
        - 'statement bindColumns' ... 'Operation '+' not understood:'
        - 'statement bindColumns' ... ' system primitive failed: '' <= 8192'
    - Error retries on **ODBC** calls
    - **Mail Queue** ordering fix
    - **Mail Macro** new **type** parameter
    - **Domain Recognition** from certain browsers
    - **Directory Browsing** within a Domain

- new **Directory Browsing** arguments
- **Call macro** and the **%cmd%** variable
- **Startup file** may be in a **NOACCESS** directory
- Added **dynamic search** to the **Insert** macro
- Added **network filename** recognition to the **WriteFile** macro
- Fixed **Data Sources** recognition problem in **ExperForms** **(Build 67)**

- **WebWizard Upgrades**

Build 67 Release Notes, September 2001

Agent3W

**These notes are still under construction ...
additional data will be added in the near future.
For more information, please contact WebBase Sales.**

- What is Agent3W?
- Get macro
- Post macro
- string functions to help parse web pages

Return to top of this page. Return to the Release Notes Main Page.

What is Agent3W?

Agent3W is a **WebBase** ***add-on*** that allows one to write web forms that access external web sites just as a browser would, returning the HTML data (or image, audio, etc.) that was referenced. ***A valid Licensed Features key for Agent3W is required to utilize its features.***

Browsers reference web pages using either a **GET** or a **POST** HTTP operation. **WebBase** with **Agent3W** emulates these operations by means of **Get** and **Post** macros. These macros reference the external web pages to be referenced, encapsulate any arguments that are required in the proper format, and return the resulting data in the specified **WebBase** variable. The contents of that variable can then be stored, parsed, and/or displayed to the user as part of the **WebBase** form that was processed depending upon the design of the form.

> **Note:** Many web pages contain copyrighted material. It is the responsibility of the **WebBase** form designer to utilize **Agent3W** responsibily. **WebBase Inc.** assumes no liability for the way in which its **Agent3W** product is utilized.

Return to top of this page. Return to the Release Notes Main Page.

Get macro

The **get** macro sends a **GET** request to the specified URL and receives the reply from the host as if a browser were issuing the request.

Both the **get** and the **post** macros have two types of arguments that can be sent with their request; those that comprise the *header* that is associated with the command, usually decided by the browser, and those that are generated in some fashion by choices a user makes at the browser. Header arguments are hidden from the user while values that the user might be asked to provide would be sent as part of the command line with the **GET** or appended to the header for a **POST**.

Both macros also allow the user to indicate how the returned data is to be handled. If no specification is made, the returned data is simply inserted into the output stream replacing the entire **{get ...} ... {/get}** (or **post**) form. If the user suplies a variable to accept the returned data, then nothing is returned to the output stream but rather is stored in that variable.

The **get** macro is almost identical to the **post** macro mentioned below - all macro variables and arguments are the same. The only difference is that when issuing the request to the remote host, a **GET** command is specified instead of a **POST** command, and any arguments specified are formatted in **GET** command format by being appended to the command line itself.

E.g.,

```
{get url 'http://www.somesite.com/somedir/somepage.htm' to replyVar
     page pageVar    header headerVar    error errorVar    args userArgs
     idle idle    sendheader sendHeader    authorization authorization }
    arg1 = 'value1'
    arg2 = 'value2'
  {/get}
```

Macro variables:

tag	value
url*	The ***url*** variable specifies a fully-qualified "http" address as would be specified in a browser's location field. E.g., the string 'http://www.webbase.com' is a valid url value. This value may also be provided by a **WebBase** variable that contains such a string.
to	The optional ***to*** argument identifies a variable to contain the returned HTML. This HTML data is returned in the form a text string and contains the data returned from the openning <HTML> tag through the closing </HTML> tag. Header information is removed before storing the web page in this variable. If this optional ***to*** argument is not included the returned HTML is placed into the output stream to be returned to the browser accessing this form.
page	The optional ***page*** argument identifies a variable to contain the entire returned text including both header and web page data. This argument may be specified in place of the above ***to*** argument or along with it. Note that using only this ***page*** argument will not prevent the web page text being returned to the browser as the ***to*** argument does.
header	The optional ***header*** argument identifies a variable to contain only the header portion of the returned text - this is the header information that the targetted web server is returning to what it believes to be a browser

visiting its site.

error The optional ***error*** argument identifies a variable to contain any error string that might be returned by the **GET** or **POST** macro. As with most other **WebBase** macros or script forms, errors in the **GET** or **POST** macro would normally be reported back to the browser. These forms could be wrapped with the **errorProtect** macro to prevent the error data from being displayed in the browser - this ***error*** argument serves the same purpose as the **errorProtect** macro.

args The optional ***args*** argument is another means of passing arguments to the targetted URL. The ***args*** argument may specify a variable that contains either a **Dictionary** where the keys are argument names and the correponding values are the values for those arguments or it may specify a variable that contains a **Collection** of variable names where the variable names become the arguments and their values the corresponding values.

If arguments are specified both via this ***args*** argument and by the arg&nsbp;=&nsbp;value notation between the {get ...} and {/get ...} (or {post ...} and {/post ...}) tags, the arguments will be combined into a single collection of arguments and values and formatted properly for the **GET** or **POST** call.

idle The optional ***idle*** argument specifies either an integer value or a variable that resolves to an integer value defining the number of seconds of idle time to wait before returning a timeout error. This is not the total time required to request and receive a web page from the remote host but the amount of idle time between communications between **WebBase** and the remote host - requesting and receiving the web page generally takes a number of buffer transactions. If more than the indicated ***idle*** time elapses without any communications from either side a timeout error is generated by **WebBase**. When this optional argument is not specified no timeout process is initiated and **WebBase** could theoretically wait forever for a host that is not responding.

sendheader

The optional ***sendheader*** argument specifies header information that is to be sent to the remote host as part of the **GET** (or **POST**0 request. This header information should mimic the header that would be sent by a browser as most web servers that process header information will be expecting a header as formatted by common web browsers.

The header information may be provided in a number of formats:

- ***a String*** - will be taken as presented and sent to the remote host without any processing;
- ***a Dictionary*** - **WebBase** will create the header string from the dictionary data in the form of <Dictionary key>: <Dictionary value> pairs;
- ***a Collection of Associations*** - **WebBase** will create the header string from the dictionary data in the form of <Association key>: <Association value> pairs;
- ***a Collection of Variable Names*** - **WebBase** will create the

header string from the dictionary data in the form of <Variable name>: <Variable value> pairs.

If this argument is not specified **WebBase** sends a primitive default header that contains the following information:

```
Pragma: no-cache
User-Agent: Mozilla/2.0 (WinNT; I)
Connection: Keep-Alive
Accept: */*
```

authorization The optional ***authorization*** argument may be used with a remote host that requires **Basic Authentication** data. This information is provided in the form of a stirng as **UserName:Password** - i.e., the user name followed by a colon **:** and the password. When provided, **WebBase** sends this data as part of the **GET** or **POST** to the remote host.

* = required variable

Return to top of this page. Return to the Release Notes Main Page.

Post macro

The **post** macro sends a **POST** request to the specified URL and receives the reply from the host as if a browser were issuing the request.

The **post** macro is almost identical to the **get** macro described above - all macro variables and arguments are the same. The only difference is that when issuing the request to the remote host, a **POST** command is specified instead of a **GET** command, and any arguments specified are formatted in **POST** command format by being appended to the header block.

E.g.,

```
{post url 'http://www.somesite.com/somedir/somepage.htm' to replyVar
     page pageVar    header headerVar    error errorVar    args userArgs
     idle idle    sendheader sendHeader    authorization authorization }
    arg1 = 'value1'
    arg2 = 'value2'
  {/post}
```

Return to top of this page. Return to the Release Notes Main Page.

Miscellaneous string functions to help parse web pages

There are a number of functions written on **String** that are designed to aid in parsing the text that is returned from a remote host. **WebBase** does not implement a true HTML parser in any sense but does have a number of string parsing and searching functions that allow one to extract data from a returned web page.

Return to top of this page. Return to the Release Notes Main Page.

Build 91 Release Notes, March 2003
